# Poster: Randomness Unmasked: Towards Reproducible and Fair Evaluation of Shift-Aware Deep Learning NIDS

Lucy Steele
lucy.steele21@imperial.ac.uk
Department of Computing
Imperial College London
London, United Kingdom

Fahad Alotaibi
f.alotaibi21@imperial.ac.uk
Department of Computing
Imperial College London
London, United Kingdom

Sergio Maffeis
sergio.maffeis@imperial.ac.uk
Department of Computing
Imperial College London
London, United Kingdom

## Abstract

Deep learning techniques are increasingly being incorporated into NIDS. However, the evaluation of such deep learning models often assumes static data distributions and overlooks the effects of randomness and environmental variation. As a result, the reported performance may not reflect the NIDS behaviour during real-world deployment. This paper investigates the impact of stochastic and environmental factors on the evaluation of deep learning models for NIDS, with a focus on shift-aware models that detect and adapt to data shift, representing state-of-the-art systems for long-term deployment. We examine two baselines under controlled variations to analyse the impact of each factor on the reproducibility and fairness of the results, revealing that the $F_1$ score can vary largely due to these, even minor, variations. All of the explored factors affect the reproducibility of the results, and some can significantly skew performance. Based on our findings, we provide practical recommendations to support reproducible and fair evaluations of deep learning-based NIDS systems.

## CCS Concepts

• **Security and privacy** → **Network security**; • **Computing methodologies** → **Neural networks**; *Supervised learning*.

## Keywords

Distribution shift, Reproducibility, Network Intrusion Detection, Deep Learning.

**ACM Reference Format:**
Lucy Steele, Fahad Alotaibi, and Sergio Maffeis. 2025. Poster: Randomness Unmasked: Towards Reproducible and Fair Evaluation of Shift-Aware Deep Learning NIDS. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25), October 13–17, 2025, Taipei, Taiwan.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3719027.3760743

## 1 Introduction

Deep Neural Networks (DNNs) power some of the most accurate Network Intrusion Detection Systems (NIDS) available today, often achieving $F_1$ scores above 95% in closed-world environments [2]. However, these detectors learn decision boundaries from a single snapshot of network traffic. As the protected network evolves through new applications, routing policies, or attacker tooling, the underlying data distribution shifts and accuracy declines. Consequently, a modern NIDS must be shift-aware: it must detect distribution changes, adapt rapidly, and do so with a limited budget for annotation by human analysts. A common strategy is a three-stage control loop: first, identify that a shift has occurred; second, query a budget-constrained subset of flows for labelling by analysts; and third, fine-tune the model on the newly labelled set. Recent frameworks such as `CADE` [6] and `INSOMNIA` [1] address parts of this loop, but they are typically evaluated with a single run, implicitly assuming deterministic performance.

It is known that DNNs exhibit substantial variance due to hidden factors, including random seeds, weight initialisation, hyperparameters, hardware, framework implementation, and determinism flags [4]. In a shift-aware NIDS, several stages are chained (shift detection, sample selection, finetuning [1, 6]), so small variances can cascade and amplify, yet their joint impact has never been measured. We close this gap with the first systematic variance audit of shift-aware DNN-based NIDS: a factorial study over the 6 main variance sources (see Table 1) spanning 630 controlled runs.

## 2 Methodology

In this preliminary study we focus on two representative, state-of-the-art, DNN-based baselines.

The first baseline is `INSOMNIA` [1], an uncertainty-sampling and pseudo-labelling framework evaluated on the 5 days of CICIDS-2017 [3]. Days 1-2 are used for pre-training; days 3–5 introduce staged attacks for adaptation.

The second baseline is `CADE` [6], a contrastive-learning detector targeting novel classes, evaluated on CSE-CIC-IDS-2018 [5]. Three one-class-held-out experiments (SSH-Patator, DoS Hulk, Infiltration) are averaged, following the original protocol.

Table 1 lists the six sources of variablity we manipulate. The ten training *seeds* `[57,305,5,9667,405,750,1038,840,63,988]` are reused across all experiments. To isolate weight initialisation noise, every run is repeated with three fixed Xavier-uniform schemes applied to layers 1 to 3 and the output layer, using seed tuples `[1004,77,259,35]`, `[8,358,200,35]`, and `[487,22,900,7]`.

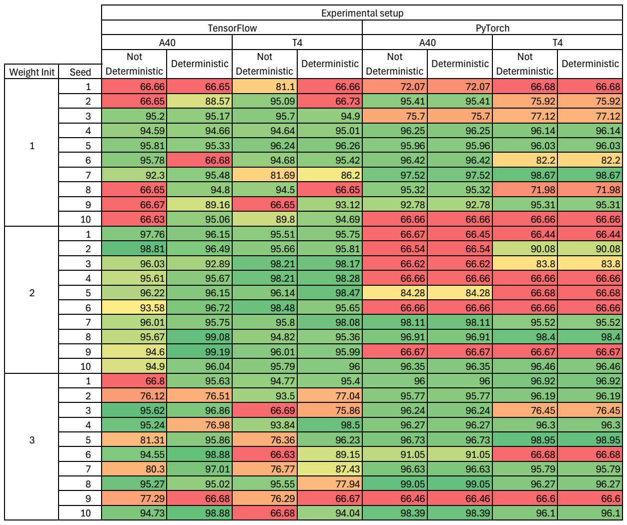
## 3 Results

The overall performance variation across the complete factorial design is visualised in Figure 1 for `CADE` and Figure 2 for `INSOMNIA`. We first outline the main trends revealed by these aggregates, then examine the influence of each individual factor in detail.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
*CCS '25, Taipei, Taiwan*
© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1525-9/2025/10
https://doi.org/10.1145/3719027.3760743

**Table 1: Controlled factors and levels.**

| Factor | Levels / Notes |
|---|---|
| Training seed ($s$) | 10 values |
| Weight initialisation ($w$) | 3 tuples |
| Model Architecture ($a$) | 2 fixed sets from independent hyperparameter searches |
| GPU model ($g$) | NVIDIA Tesla T4, A40, A100-MIG |
| Framework ($f$) | TensorFlow 2.10 vs. PyTorch 2.1 |
| Deterministic ($d$) | `TF_DETERMINISTIC_OPS`, `TF_CUDNN_DETERMINISM`, or `off`; PyTorch uses `torch.use_deterministic_algorithms(True)` etc. |



| | | Experimental setup | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | TensorFlow | | | | PyTorch | | | |
| | | A40 | | T4 | | A40 | | T4 | |
| Weight Init | Seed | Not Deterministic | Deterministic | Not Deterministic | Deterministic | Not Deterministic | Deterministic | Not Deterministic | Deterministic |
| 1 | 1 | 66.66 | 66.65 | 81.1 | 66.66 | 72.07 | 72.07 | 66.68 | 66.68 |
| | 2 | 66.65 | 88.57 | 95.09 | 66.73 | 95.41 | 95.41 | 75.92 | 75.92 |
| | 3 | 95.2 | 95.17 | 95.7 | 94.9 | 75.7 | 75.7 | 77.12 | 77.12 |
| | 4 | 94.59 | 94.66 | 94.64 | 95.01 | 96.25 | 96.25 | 96.14 | 96.14 |
| | 5 | 95.81 | 95.33 | 96.24 | 96.26 | 95.96 | 95.96 | 96.03 | 96.03 |
| | 6 | 95.78 | 66.68 | 94.68 | 95.42 | 96.42 | 96.42 | 82.2 | 82.2 |
| | 7 | 92.3 | 95.48 | 81.69 | 86.2 | 97.52 | 97.52 | 98.67 | 98.67 |
| | 8 | 66.65 | 94.8 | 94.5 | 66.65 | 95.32 | 95.32 | 71.98 | 71.98 |
| | 9 | 66.67 | 89.16 | 66.65 | 93.12 | 92.78 | 92.78 | 95.31 | 95.31 |
| | 10 | 66.63 | 95.06 | 89.8 | 94.69 | 66.66 | 66.66 | 66.66 | 66.66 |
| 2 | 1 | 97.76 | 96.15 | 95.51 | 95.75 | 66.67 | 66.45 | 66.44 | 66.44 |
| | 2 | 98.81 | 96.49 | 95.66 | 95.81 | 66.54 | 66.54 | 90.08 | 90.08 |
| | 3 | 96.03 | 92.89 | 98.21 | 98.17 | 66.62 | 66.62 | 83.8 | 83.8 |
| | 4 | 95.61 | 95.67 | 98.21 | 98.28 | 66.66 | 66.66 | 66.66 | 66.66 |
| | 5 | 96.22 | 96.15 | 96.14 | 98.47 | 84.28 | 84.28 | 66.68 | 66.68 |
| | 6 | 93.58 | 96.72 | 98.48 | 95.65 | 66.66 | 66.66 | 66.66 | 66.66 |
| | 7 | 96.01 | 95.75 | 95.8 | 98.08 | 98.11 | 98.11 | 95.52 | 95.52 |
| | 8 | 95.67 | 99.08 | 94.82 | 95.36 | 96.91 | 96.91 | 98.4 | 98.4 |
| | 9 | 94.6 | 99.19 | 96.01 | 95.99 | 66.67 | 66.67 | 66.67 | 66.67 |
| | 10 | 94.9 | 96.04 | 95.79 | 96 | 96.35 | 96.35 | 96.46 | 96.46 |
| 3 | 1 | 66.8 | 95.63 | 94.77 | 95.4 | 96 | 96 | 96.92 | 96.92 |
| | 2 | 76.12 | 76.51 | 93.5 | 77.04 | 95.77 | 95.77 | 96.19 | 96.19 |
| | 3 | 95.62 | 96.86 | 66.69 | 75.86 | 96.24 | 96.24 | 76.45 | 76.45 |
| | 4 | 95.24 | 76.98 | 93.84 | 98.5 | 96.27 | 96.27 | 96.3 | 96.3 |
| | 5 | 81.31 | 95.86 | 76.36 | 96.23 | 96.73 | 96.73 | 98.95 | 98.95 |
| | 6 | 94.55 | 98.88 | 66.63 | 89.15 | 91.05 | 91.05 | 66.68 | 66.68 |
| | 7 | 80.3 | 97.01 | 76.77 | 87.43 | 96.63 | 96.63 | 95.79 | 95.79 |
| | 8 | 95.27 | 95.02 | 95.55 | 77.94 | 99.05 | 99.05 | 96.27 | 96.27 |
| | 9 | 77.29 | 66.68 | 76.29 | 66.67 | 66.46 | 66.46 | 66.6 | 66.6 |
| | 10 | 94.73 | 98.88 | 66.68 | 94.04 | 98.39 | 98.39 | 96.1 | 96.1 |

**Figure 1: $F_1$ scores for all of our experimental setups and seeding combinations on CADE.**

**Table 2: Aggregated $F_1$ ranges across *all* experimental factors.**

| | Minimum | Mean | Maximum |
|---|---|---|---|
| INSOMNIA | 65.28 | 80.99 | 89.42 |
| CADE | 66.44 | 87.15 | 99.19 |

## 3.1 Aggregated performance distributions

Across all 630 controlled runs (390 INSOMNIA, 240 CADE) the $F_1$ varies dramatically (Table 2). For example, in one setting, INSOMNIA scores as low as 65.28% under a combination of factors but climbs to almost 90% under another. This 24-point swing highlights how seemingly minor choices in randomness and environment can dominate the headline results, and need to be controlled to ensure reproducibility and objective model comparison.

## 3.2 Factor analysis

*3.2.1 Hyperparameters.* To isolate architectural effects we repeated the nondeterministic hyperparameter search described in INSOMNIA twice, yielding two distinct configurations, $\mathbf{H}_1$ and $\mathbf{H}_2$. Each was trained with the full randomness sweep: ten training seeds $s \in S$ crossed with the three weight initialisation tuples $w \in W$ ($|S| \times |W| = 30$ runs per configuration). Table 3 reports the aggregated $F_1$ distributions. Switching from $\mathbf{H}_1$ to $\mathbf{H}_2$ shifts the mean $F_1$ by up to 9%, showing that seemingly minor architectural tweaks can dominate overall performance and therefore must be disclosed in any fair comparison.

**Table 3: Effect of hyperparameter sets on INSOMNIA (T4 + TensorFlow + deterministic).**

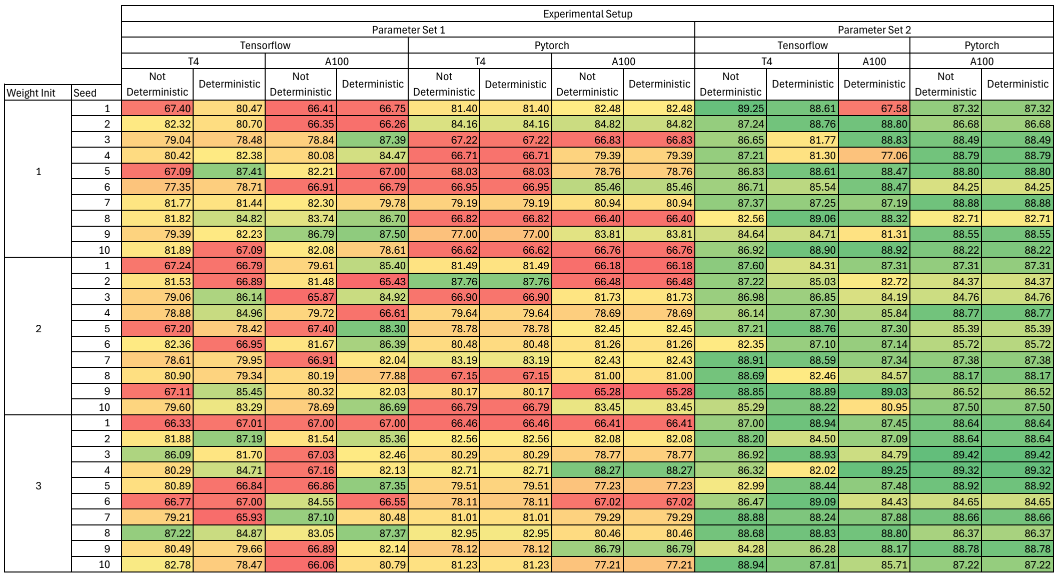
| | Minimum | Mean | Maximum |
|---|---|---|---|
| $\mathbf{H}_1$ | 66.33 | 77.76 | 87.22 |
| $\mathbf{H}_2$ | 82.35 | 86.78 | 89.25 |
| $\Delta$ | +16.02 | +9.02 | +2.03 |

*3.2.2 Weight initialisation.* Although weight initialisation is rarely documented, our results show it can be as influential as the training seed. To isolate its effect we freeze all other factors and vary only the weight factor $w$. Even this single change produces large swings: on CADE running deterministic TensorFlow on a T4 GPU, the $F_1$ leaps from 66.66 % for ($s$=1, $w_1$) to 95.75 % for ($s$=1, $w_2$) (Figure 1). Averaged over all ten seeds, the same setup shifts the mean from 88.16 % ($w_1$) to 96.41 % ($w_2$). These findings underline that $w$ must be reported and swept alongside the conventional training seed to ensure fair and reproducible comparisons.

*3.2.3 GPU model.* Because independent researchers replicating baselines will often do so on whatever heterogeneous GPUs they have at hand, we run our randomness tests on two distinct graphics cards to see whether a change in the card biases the results. With a single run the gap can appear large. For example, CADE on an A40 (nondeterministic TensorFlow, $w_1$, $s$ = 2) reports a noticeably higher score than the same code on a T4. Once we average across multiple runs, however, the effect nearly disappears. For INSOMNIA (deterministic TensorFlow) the mean $F_1$ score is 78.18% on a T4 compared with 79.29% on an A100, a relative difference of just 1.1%. CADE tells a similar story: 91.47% on an A40 versus 89.38% on a T4, a 2.1% gap. In short, according to our experiments, run-to-run variance overwhelms any GPU-specific bias, and averaging only a few repetitions is enough to neutralise hardware effects.

*3.2.4 Frameworks.* The choice of deep learning framework is itself a confounding variable, so we evaluate every baseline in the field's two dominant frameworks: TensorFlow and PyTorch. A one-off measurement is unreliable: on an NVIDIA T4, CADE in deterministic TensorFlow ($w$=1, $s$=3) scores 94.9%, whereas the same configuration in PyTorch manages only 77.2%. Repeating the experiment over seed and weight intialisation values and averaging attenuates the discrepancy: TensorFlow stabilises at 91.47% and PyTorch at 86.40%, for a difference of ≈ 5%. INSOMNIA shows a similar pattern (79.29% vs. 77.60%). Consequently, cross-framework comparisons are acceptable when a 5% swing is tolerable, but only if results are aggregated over multiple runs; otherwise, any single figure is apt to mislead.

*3.2.5 Determinism.* We isolate the impact of enabling deterministic operations in deep learning frameworks (factor $d$), a practice commonly recommended for reproducibility. As with other environmental factors, individual trial results exhibit considerable variance,



| | | Experimental Setup | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Parameter Set 1 | | | | | | | | Parameter Set 2 | | | | |
| | | Tensorflow | | | | Pytorch | | | | Tensorflow | | | Pytorch | |
| | | T4 | | A100 | | T4 | | A100 | | T4 | | A100 | A100 | |
| Weight Init | Seed | Not Deterministic | Deterministic | Not Deterministic | Deterministic | Not Deterministic | Deterministic | Not Deterministic | Deterministic | Not Deterministic | Deterministic | Deterministic | Not Deterministic | Deterministic |
| 1 | 1 | 67.40 | 80.47 | 66.41 | 66.75 | 81.40 | 81.40 | 82.48 | 82.48 | 89.25 | 88.61 | 67.58 | 87.32 | 87.32 |
| | 2 | 82.32 | 80.70 | 66.35 | 66.26 | 84.16 | 84.16 | 84.82 | 84.82 | 87.24 | 88.76 | 88.80 | 86.68 | 86.68 |
| | 3 | 79.04 | 78.48 | 78.84 | 87.39 | 67.22 | 67.22 | 66.83 | 66.83 | 86.65 | 81.77 | 88.83 | 88.49 | 88.49 |
| | 4 | 80.42 | 82.38 | 80.08 | 84.47 | 66.71 | 66.71 | 79.39 | 79.39 | 87.21 | 81.30 | 77.06 | 88.79 | 88.79 |
| | 5 | 67.09 | 87.41 | 82.21 | 67.00 | 68.03 | 68.03 | 78.76 | 78.76 | 86.83 | 88.61 | 88.47 | 88.80 | 88.80 |
| | 6 | 77.35 | 78.71 | 66.91 | 66.79 | 66.95 | 66.95 | 85.46 | 85.46 | 86.71 | 85.54 | 88.47 | 84.25 | 84.25 |
| | 7 | 81.77 | 81.44 | 82.30 | 79.78 | 79.19 | 79.19 | 80.94 | 80.94 | 87.37 | 87.25 | 87.19 | 88.88 | 88.88 |
| | 8 | 81.82 | 84.82 | 83.74 | 86.70 | 66.82 | 66.82 | 66.40 | 66.40 | 82.56 | 89.06 | 88.32 | 82.71 | 82.71 |
| | 9 | 79.39 | 82.23 | 86.79 | 87.50 | 77.00 | 77.00 | 83.81 | 83.81 | 84.64 | 84.71 | 81.31 | 88.55 | 88.55 |
| | 10 | 81.89 | 67.09 | 82.08 | 78.61 | 66.62 | 66.62 | 66.76 | 66.76 | 86.92 | 88.90 | 88.92 | 88.22 | 88.22 |
| 2 | 1 | 67.24 | 66.79 | 79.61 | 85.40 | 81.49 | 81.49 | 66.18 | 66.18 | 87.60 | 84.31 | 87.31 | 87.31 | 87.31 |
| | 2 | 81.53 | 66.89 | 81.48 | 65.43 | 87.76 | 87.76 | 66.48 | 66.48 | 87.22 | 85.03 | 82.72 | 84.37 | 84.37 |
| | 3 | 79.06 | 86.14 | 65.87 | 84.92 | 66.90 | 66.90 | 81.73 | 81.73 | 86.98 | 86.85 | 84.19 | 84.76 | 84.76 |
| | 4 | 78.88 | 84.96 | 79.72 | 66.61 | 79.64 | 79.64 | 78.69 | 78.69 | 86.14 | 87.30 | 85.84 | 88.77 | 88.77 |
| | 5 | 67.20 | 78.42 | 67.40 | 88.30 | 78.78 | 78.78 | 82.45 | 82.45 | 87.21 | 88.76 | 87.30 | 85.39 | 85.39 |
| | 6 | 82.36 | 66.95 | 81.67 | 86.39 | 80.48 | 80.48 | 81.26 | 81.26 | 82.35 | 87.10 | 87.14 | 85.72 | 85.72 |
| | 7 | 78.61 | 79.95 | 66.91 | 82.04 | 83.19 | 83.19 | 82.43 | 82.43 | 88.91 | 88.59 | 87.34 | 87.38 | 87.38 |
| | 8 | 80.90 | 79.34 | 80.19 | 77.88 | 67.15 | 67.15 | 81.00 | 81.00 | 88.69 | 82.46 | 84.57 | 88.17 | 88.17 |
| | 9 | 67.11 | 85.45 | 80.32 | 82.03 | 80.17 | 80.17 | 65.28 | 65.28 | 88.85 | 88.89 | 89.03 | 86.52 | 86.52 |
| | 10 | 79.60 | 83.29 | 78.69 | 86.69 | 66.79 | 66.79 | 83.45 | 83.45 | 85.29 | 88.22 | 80.95 | 87.50 | 87.50 |
| 3 | 1 | 66.33 | 67.01 | 67.00 | 67.00 | 66.46 | 66.46 | 66.41 | 66.41 | 87.00 | 88.94 | 87.45 | 88.64 | 88.64 |
| | 2 | 81.88 | 87.19 | 81.54 | 85.36 | 82.56 | 82.56 | 82.08 | 82.08 | 88.20 | 84.50 | 87.09 | 88.64 | 88.64 |
| | 3 | 86.09 | 81.70 | 67.03 | 82.46 | 80.29 | 80.29 | 78.77 | 78.77 | 86.92 | 88.93 | 84.79 | 89.42 | 89.42 |
| | 4 | 80.29 | 84.71 | 67.16 | 82.13 | 82.71 | 82.71 | 88.27 | 88.27 | 86.32 | 82.02 | 89.25 | 89.32 | 89.32 |
| | 5 | 80.89 | 66.84 | 66.86 | 87.35 | 79.51 | 79.51 | 77.23 | 77.23 | 82.99 | 88.44 | 87.48 | 88.92 | 88.92 |
| | 6 | 66.77 | 67.00 | 84.55 | 66.55 | 78.11 | 78.11 | 67.02 | 67.02 | 86.47 | 89.09 | 84.43 | 84.65 | 84.65 |
| | 7 | 79.21 | 65.93 | 87.10 | 80.48 | 81.01 | 81.01 | 79.29 | 79.29 | 88.88 | 88.24 | 87.88 | 88.66 | 88.66 |
| | 8 | 87.22 | 84.87 | 83.05 | 87.37 | 82.95 | 82.95 | 80.46 | 80.46 | 88.68 | 88.83 | 88.80 | 86.37 | 86.37 |
| | 9 | 80.49 | 79.66 | 66.89 | 82.14 | 78.12 | 78.12 | 86.79 | 86.79 | 84.28 | 86.28 | 88.17 | 88.78 | 88.78 |
| | 10 | 82.78 | 78.47 | 66.06 | 80.79 | 81.23 | 81.23 | 77.21 | 77.21 | 88.94 | 87.81 | 85.71 | 87.22 | 87.22 |

**Figure 2: $F_1$ scores for all of our experimental setups and seeding combinations on `INSOMNIA`.**

but averaging across runs mitigates this effect. On an NVIDIA A100, `INSOMNIA` in TensorFlow improves from an average $F_1$ of 75.83 to 79.29 when determinism is enabled (+3.46%). Similarly, for `CADE` on an A40, the mean increases from 87.45 to 91.47 (+4.02%). These results highlight the importance of reporting whether deterministic operations are enabled, as the averaged performance can vary by up to 4%.

## 4 Conclusions

We have shown that seemingly minor implementation choices can change shift-aware DNN-based NIDS $F_1$ scores by up to ≈33 percentage points (*e.g.*, `CADE` from 66.44% to 99.19%), often eclipsing the improvements attributed to new algorithms. Single-architecture and single-seed evaluations, though convenient, are inadequate and risk misrepresenting performance. We therefore advocate a variance-aware evaluation protocol:

(1) Report distributions (min/mean/max/std) over ≥10 seeds and 3 weight initializations; single runs are misleading. As a rule of thumb, 15 runs mixing seeds and initializations approximate the full distribution (average error < 2 points, 95th percentile ≈4), balancing accuracy and cost.
(2) Record GPU model and enable framework-level determinism for reproducibility. In our experiments, determinism also modestly improves average performance.
(3) If model hyperparameter selection is non-deterministic, test multiple configurations; a single choice can mask strengths and weaknesses.

Adopting this protocol, which reports distributions across seeds, initializations, and architectures, is essential for trustworthy and reproducible benchmarking going forward. Beyond research, we recommend security operations teams perform pre-deployment variance audits on their own infrastructure to avoid misleading expectations and identify the most reliable model.

**Future Work.** We plan to extend this study to additional shift-aware DNN-based NIDS baselines and to examine the impact of randomness in other security domains, such as malware detection, where similar issues are likely to arise. We also aim to explore the feasibility of community-driven benchmarking platforms that programmatically control these factors, enabling fairer and more reliable comparisons across frameworks.

## 5 Acknowledgment

Alotaibi's research is funded by a scholarship from the Deanship of Scientific Research at Najran University, Kingdom of Saudi Arabia.

## References

[1] Giuseppina Andresini, Feargus Pendlebury, Fabio Pierazzi, Corrado Loglisci, Annalisa Appice, and Lorenzo Cavallaro. 2021. Insomnia: Towards concept-drift robustness in network intrusion detection. In *Proceedings of the 14th ACM workshop on artificial intelligence and security*. 111–122.
[2] Ramya Chinnasamy, Malliga Subramanian, Sathishkumar Veerappampalayam Easwaramoorthy, and Jaehyuk Cho. 2025. Deep learning-driven methods for network-based intrusion detection systems: A systematic review. *ICT Express* (2025).
[3] Gints Engelen, Vera Rimmer, and Wouter Joosen. 2021. Troubleshooting an intrusion detection dataset: the CICIDS2017 case study. In *2021 IEEE Security and Privacy Workshops (SPW)*. IEEE, 7–12.
[4] Yilin Ji, Daniel Kaestner, Oliver Wirth, and Christian Wressnegger. 2023. Randomness is the root of all evil: more reliable evaluation of deep active learning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 3943–3952.
[5] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. 2018. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy, ICISSP 2018, Funchal, Madeira - Portugal, January 22-24, 2018*. SciTePress, 108–116. doi:10.5220/0006639801080116
[6] Limin Yang, Wenbo Guo, Qingying Hao, Arridhana Ciptadi, Ali Ahmadzadeh, Xinyu Xing, and Gang Wang. 2021. CADE: Detecting and Explaining Concept Drift Samples for Security Applications. In *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*. USENIX Association, 2327–2344.